

redux vs alt

objectives

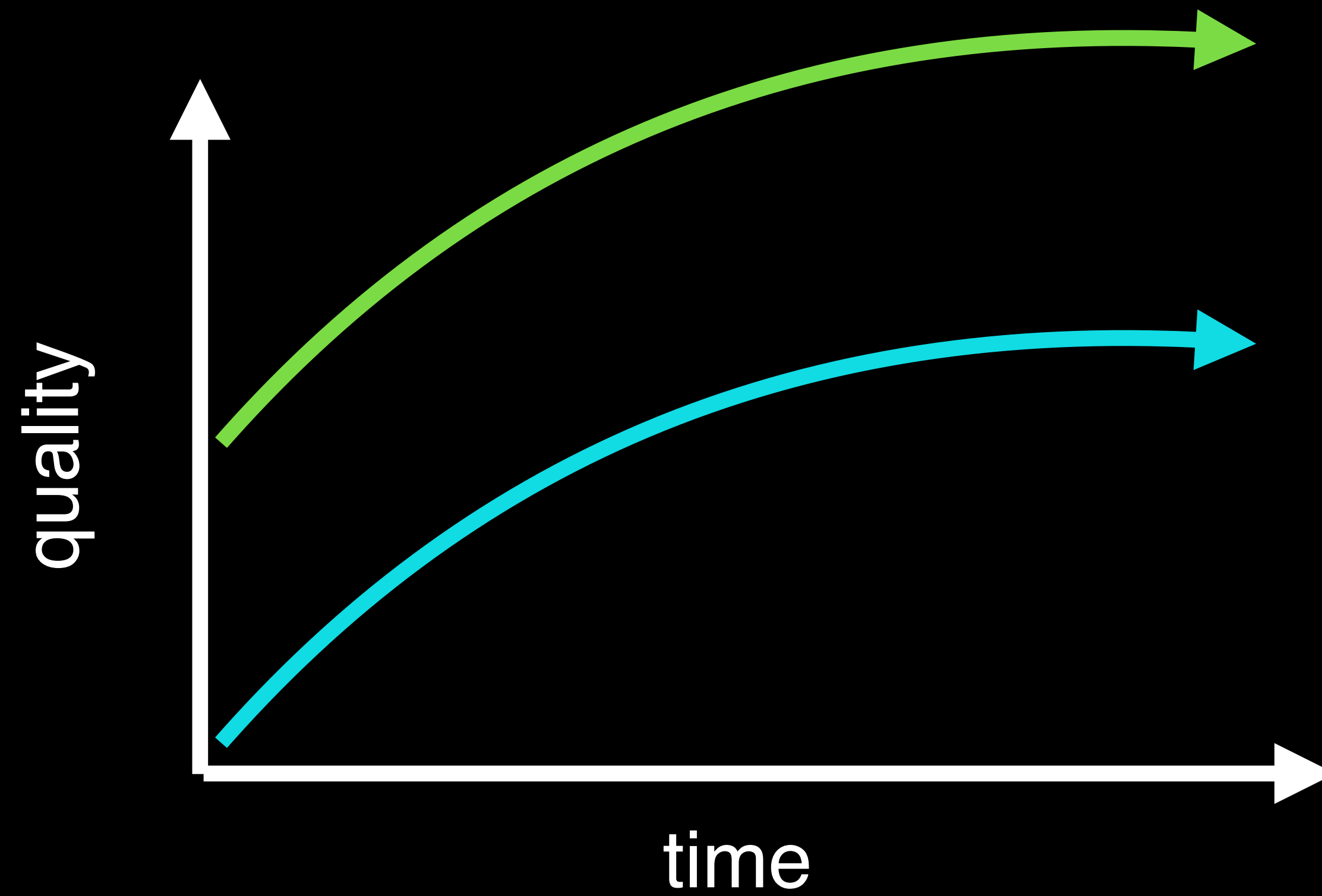
compare redux and alt:

- concepts
- testing
- trends

agenda

- react
- flux
- why?
- redux and alt comparison
- microservices for client side

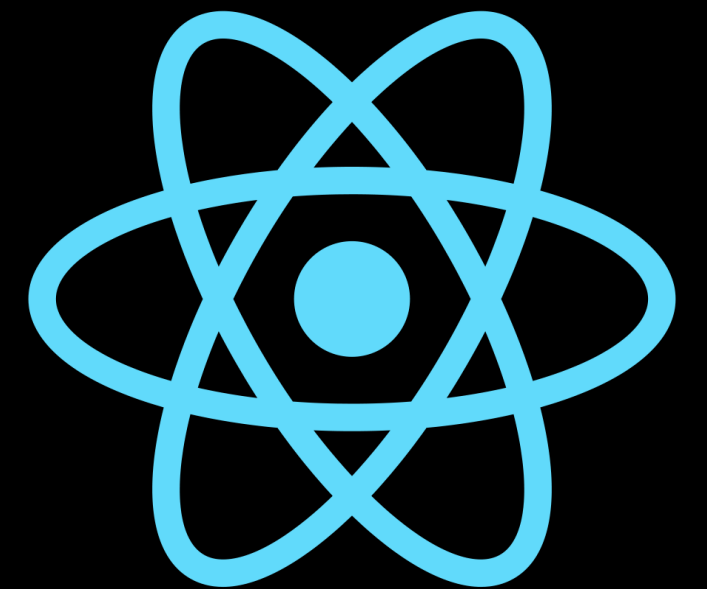
why?



quality takes time

react

- just the UI
- virtual DOM
- one way data flow

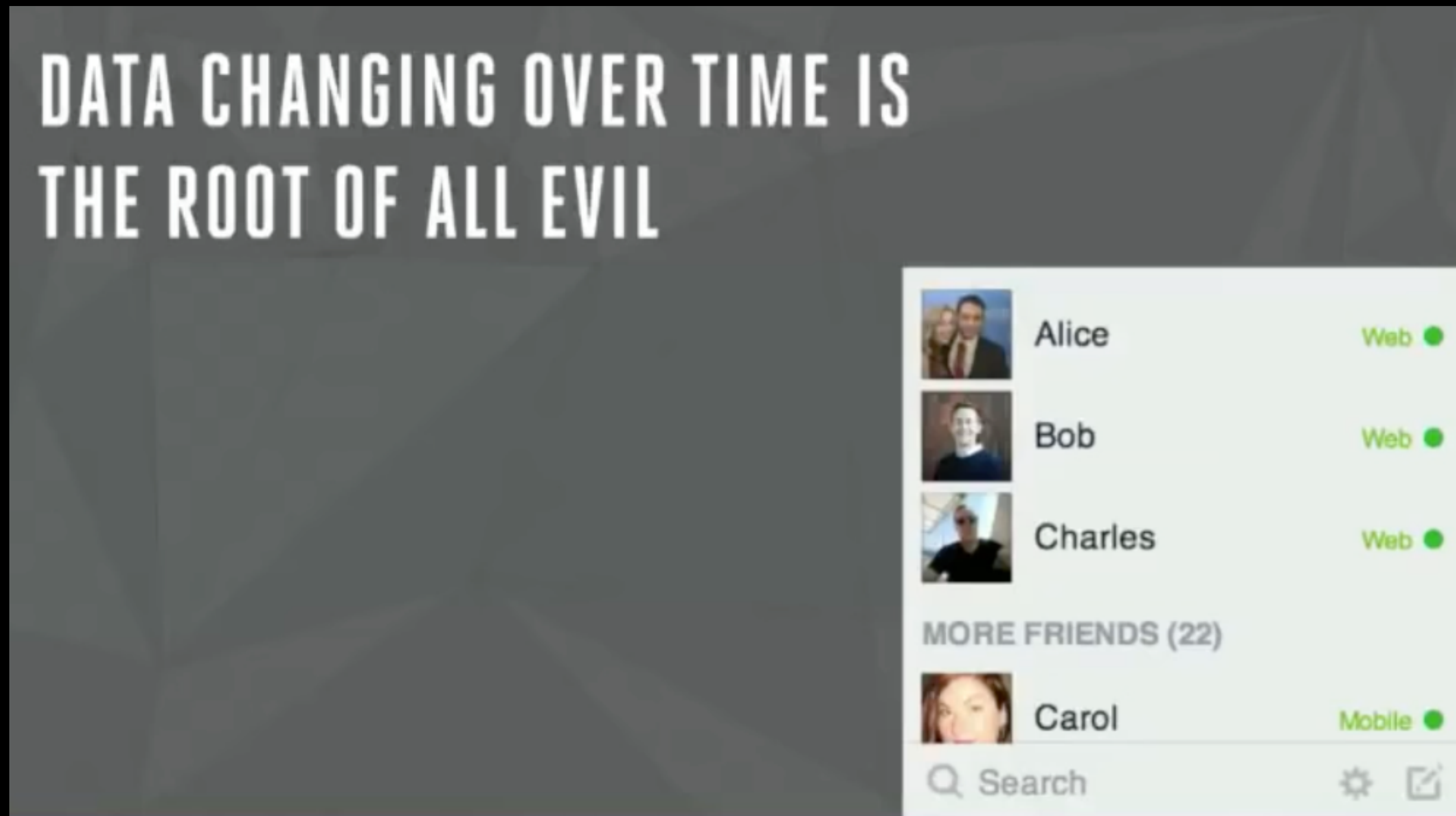


```
var HelloMessage = React.createClass({  
  render: function() {  
    return <div>Hello {this.props.name}</div>;  
  }  
});
```

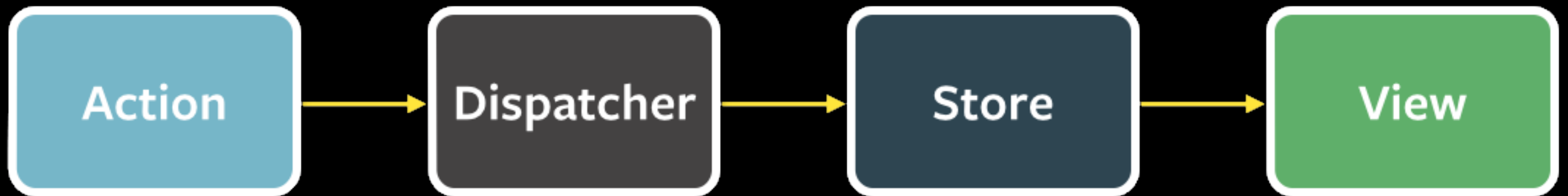
```
ReactDOM.render(<HelloMessage name="John" />, mountNode);
```

Hello John

fb contact list

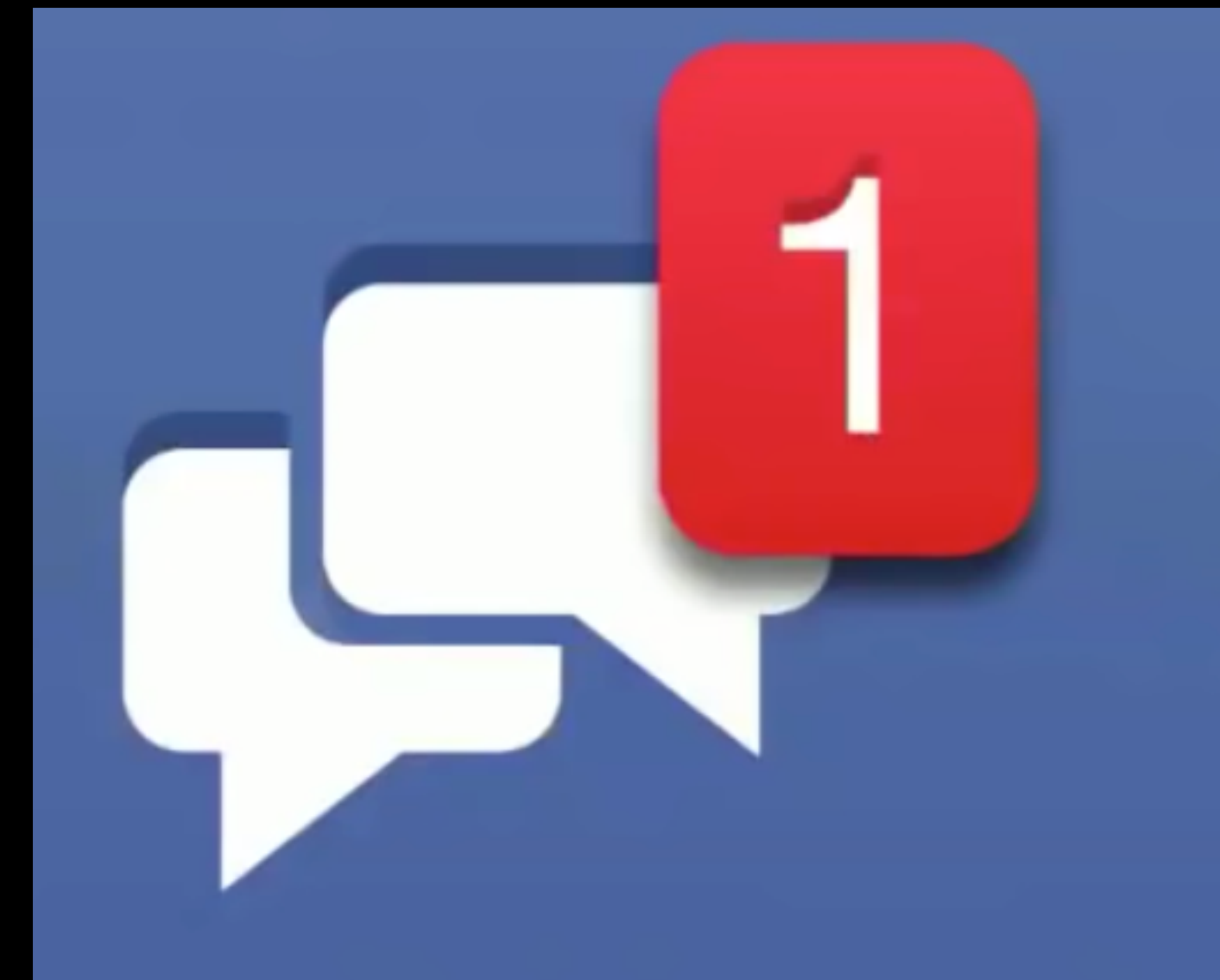
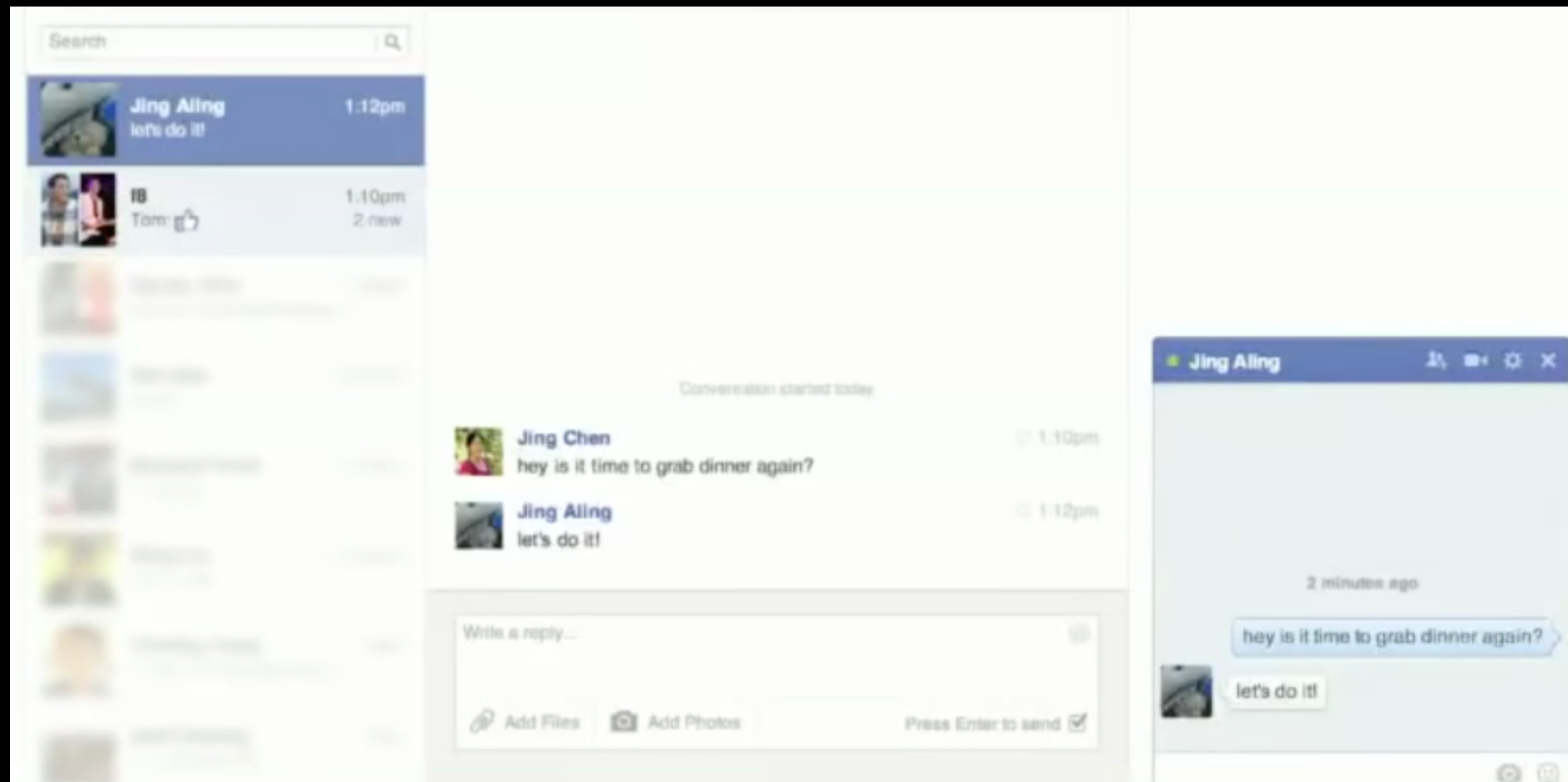


flux



Flux is the application architecture that Facebook uses for building client-side web applications. It complements React's and is more of a pattern than a framework.

fb chat



redux and alt

- implementations of flux
- isomorphic - (equal shape when rendered both on client and server side)
- support react-native

redux

predictable state container for JavaScript apps

- whole state of app is stored in an object tree inside a **single store**
- only way to change the state tree is to **emit an action**, an object describing what happened
- to specify how the actions transform the state tree, you write **pure reducers**
- great **documentation** - not that it needs one really

alt

Isomorphic flux implementation

- pure flux
- *actively* maintained
- extremely flexible and *unopinionated* in how you use flux
- has logo
- lacks up to date documentation and samples



hello world app

redux

Hello, Alpha
Hello, Gamma

- └─ app.jsx
- └─ components
 - └─ greet.jsx
 - └─ hello.jsx
 - └─ hello.test.js
 - └─ helloList.jsx
- └─ index.html
- └─ index.js
- └─ state
 - └─ namesReducer.js
 - └─ namesReducer.test.js
 - └─ store.js

alt

Hello, Alpha
Hello, Gamma

- └─ actions
 - └─ NamesActions.js
- └─ alt.js
- └─ components
 - └─ greet.jsx
 - └─ hello.jsx
 - └─ hello.test.js
 - └─ helloList.jsx
- └─ index.html
- └─ index.js
- └─ state
 - └─ configureNamesStore.js
 - └─ configureNamesStore.test.js
 - └─ namesStore.js

components

- similar or equal setup for tiny components
- differs for larger components that contain actions and more logic
- testing is easy using **react test utils** or wrapper libraries
- **redux** - allows function syntax, **alt** - needs a class that extends `React.Component` for

components - code

code

```
import React from 'react';

const Hello = ({name}) => {
  return (
    <div>
      Hello, <b class='test-name'>{name}</b>
    </div>
  );
};

export default Hello;
```

tests

```
describe('hello component', () => {
  it('renders name', () => {
    const output = renderHello('Steve');
    const name = getChildrenByClass(output, 'test-name');
    expect(name.props.children).toEqual('Steve');
  });
  it('renders component', () => {
    const output = renderHello('Steve');
    const expected =
      <div>Hello, <b class="test-name">Steve</b></div>;
    expect(output).toEqualJSX(expected);
  });
});

const renderHello = name => {
  const renderer = TestUtils.createRenderer();
  renderer.render(<Hello name={name}/>);
  return renderer.getRenderOutput();
}
```

components - tests

```
describe('<MyComponent />', () => {
```

```
  it('renders three <Foo /> components', () => {  
    const wrapper = shallow(<MyComponent />);  
    expect(wrapper.find(Foo)).toHaveLength(3);  
  });
```

```
  it('renders an `.icon-star`', () => {  
    const wrapper = shallow(<MyComponent />);  
    expect(wrapper.find('.icon-star')).toHaveLength(1);  
  });
```

```
  it('renders children when passed in', () => {  
    const wrapper = shallow(  
      <MyComponent>  
        <div className="unique" />  
      </MyComponent>  
    );  
    expect(wrapper.contains(<div className="unique" />)).toEqual(true);  
  });
```

```
  it('simulates click events', () => {  
    const onClick = sinon.spy();  
    const wrapper = shallow(  
      <Foo onClick={onClick} />  
    );  
    wrapper.find('button').simulate('click');  
    expect(onClick.calledOnce).toEqual(true);  
  });  
});
```

airbnb/enzyme

react element

css class

html element

simulate event

stores

- **redux** - one, **alt** - many
- **redux** - registers reducers, **alt** - defines logic to execute based on data passed by action
- **redux** - not tests needed, test reducers, **alt** - the meat of logic, harder to test without side effects
- **redux** - immutable, **alt** - mutable
- **alt** - ReferenceError: A store named NamesStore already exists, double check your store names or pass in your own custom identifier for each store]

reducers

- **pure functions** - zero side-effects, only rely on parameter values
- super easy to **test**
- they **scale** - easy to add a new one

react-redux glue

```
const store = configureStore();

render(
  <Provider store={store}>
    <App />
  </Provider>,
  document.getElementById( 'App' )
)
```



1

react-redux glue

```
import {connect} from 'react-redux';

const store = configureStore();

render(
  <Provider store={store}>
    <App />
  </Provider>,
  document.getElementById('App')
)

const App = ({names}) => (
  <div className='main'>
    <Greet />
    <HelloList names={names} />
  </div>
)

const mapStateToProps = state => {
  return {
    names: state.names
  }
}

export default connect(mapStateToProps)(App);
```

react-redux glue

```
const store = configureStore();

render(
  <Provider store={store}>
    <App />
  </Provider>,
  document.getElementById( 'App' )
)

import {connect} from 'react-redux';

const App = ({names}) => (
  <div className='main'>
    <Greet />
    <HelloList names={names} />
  </div>
)

const mapStateToProps = state => {
  return {
    names: state.names
  }
}

export default connect(mapStateToProps)(App);

const HelloList = ({ names }) => (
  <div className='test-hello-name-list'>
    {names.map(name =>
      <Hello name={name}/>
    )}
  </div>
)
```

react-alt glue

```
// Bad: getting into lib internals
import connectToStores from 'alt-utils/lib/connectToStores';

class HelloList extends React.Component {
  static getStores(props) {
    return [store];
  }
  static getPropsFromStores(props) {
    return store.getState();
  }
  render() {
    return (
      <div className='test-hello-name-list'>
        {this.props.names.map(name =>
          <Hello name={name}/>
        ) }
      </div>
    );
  }
}

export default connectToStores(HelloList);
```

1

react-alt glue

```
class Greet extends React.Component {
  constructor(props) {
    super(props);
    this.submit = this.submit.bind(this);
  }
  static getStores(props) {
    return [store]
  }
  static getPropsFromStores(props) {
    return store.getState();
  }
  submit(e) {
    e.preventDefault();
    namesActions.add(this.input.value);
  }
  render() {
    return (
      <div className='test-greet'>
        <form onSubmit={this.submit}>
          <input id='test-greeting' type='text' ref={node => {
            this.input = node}}/>
          <button id='test-submit-greeting' type='submit'>Greet</button>
        </form>
      </div>
    );
  }
};

export default connectToStores(Greet);
```

1

application stats

lines of code

21 ./hello-redux/components/greet.jsx
11 ./hello-redux/components/hello.jsx
32 ./hello-redux/components/hello.test.js
12 ./hello-redux/components/helloList.jsx

6 ./hello-redux/state/store.js

12 ./hello-redux/state/namesReducer.js
17 ./hello-redux/state/namesReducer.test.js

19 ./hello-redux/app.jsx
14 ./hello-redux/index.js

144 total

35 ./hello-alt/components/greet.jsx
11 ./hello-alt/components/hello.jsx
32 ./hello-alt/components/hello.test.js
25 ./hello-alt/components/helloList.jsx

20 ./hello-alt/state/configureNamesStore.js
24 ./hello-alt/state/configureNamesStore.test.js
2 ./hello-alt/state/namesStore.js

8 ./hello-alt/actions/NamesActions.js
4 ./hello-alt/alt.js
12 ./hello-alt/index.js

173 total = 20 % more

scaling application

react



alt



stats

	redux	alt
version	3.5.2	0.18.4
size	5.9K (2kb?)	23K
downloads /month	670K	38K
repo	reactjs/redux	goatslacker/alt
open issues	44	79
pull requests	20	4
last commit	hours ago	almost a month ago

comparison

redux



alt



Open Systems Don't Always Win - Steve Jobs

verdict

TRIAL

Redux is a great, mature tool that has helped many of our teams reframe how they think about managing state in client-side apps. Using a Flux-style approach, it enables a loosely coupled state-machine architecture that's easy to reason about. We've found it a good companion to some of our favored JavaScript frameworks, such as Ember and React - ThoughtWorks TechRadar 2016 April

HOLD

Alt although says to be unopinionated it pushes to use action objects, static functions that are used by connectToStores glue, possible to mutate store state that can lead to errors and having multiple stores might encourage a monolith approach, doesn't have as big community backing as Redux and feels dated in comparison

end to end tests

```
var process = require('process');
var port = process.env.PORT;

function navigateToSite(test) {
  return test.open('http://localhost:' + port);
}


module.exports = {
  'says hello': function (test) {
    navigateToSite(test)
    .assert.doesntExist('.test-name', 'initial state has no greetings')
    .type('#test-greeting', 'Steve')
    .click('#test-submit-greeting')
    .assert.text('.test-hello', 'Hello, Steve', 'adds a new greeting')
    .done();
  }
};
```


gives confidence when switching frameworks

*don't use dalekjs

example site - TES Global

[Resources](#) [Jobs](#) [Community](#) [News](#)

 [Upload](#)



This is a resource title that could possibly reach a couple of lines

[Selecting level](#) [Subject](#) [Subject](#) [Subject](#) [Subject](#)









★★★★★ 11,304 (1.0)

Files
2 movies, 5 documents [see all](#)

[Download all 5 files](#) FREE

[See more resources like this](#)

Resource and file types

	Worksheet (285Kb, Powerpoint document) Last updated: 13/04/14	
	Game, puzzle or quiz (285Kb, Word document) Last updated: 13/04/14	
	Assembly (285Kb, Powerpoint document) Last updated: 13/04/14	
	Display and posters (285Kb, Word document) Last updated: 13/04/14	

About this resource

A worksheet that can be used to test times tables. The children write whatever times table they are doing (4x, 8x etc) in the star in the middle of the sheet and times the number on the outside of the track by that number. They then write the answers of the times table in the boxes in the inside of the track. Lower ability children can be asked to +2, +4 etc to the number on the outside of the track instead of completing times tables. Give children 2 or 3 minutes to complete the race. Photocopy back to back and ask higher ability children to complete both sides. I made this worksheet during a teaching practice with Year 3 - it works well.

Comments

Brilliant resource, thank you for sharing.
from [DanaeKoe](#), 27 August 2014

Love them! Will be a very useful resource for my class.
from [m33m](#), 02 September 2012

These are fab! Thank you so much!

Rating

★★★★★

★★★★☆

★★★★☆

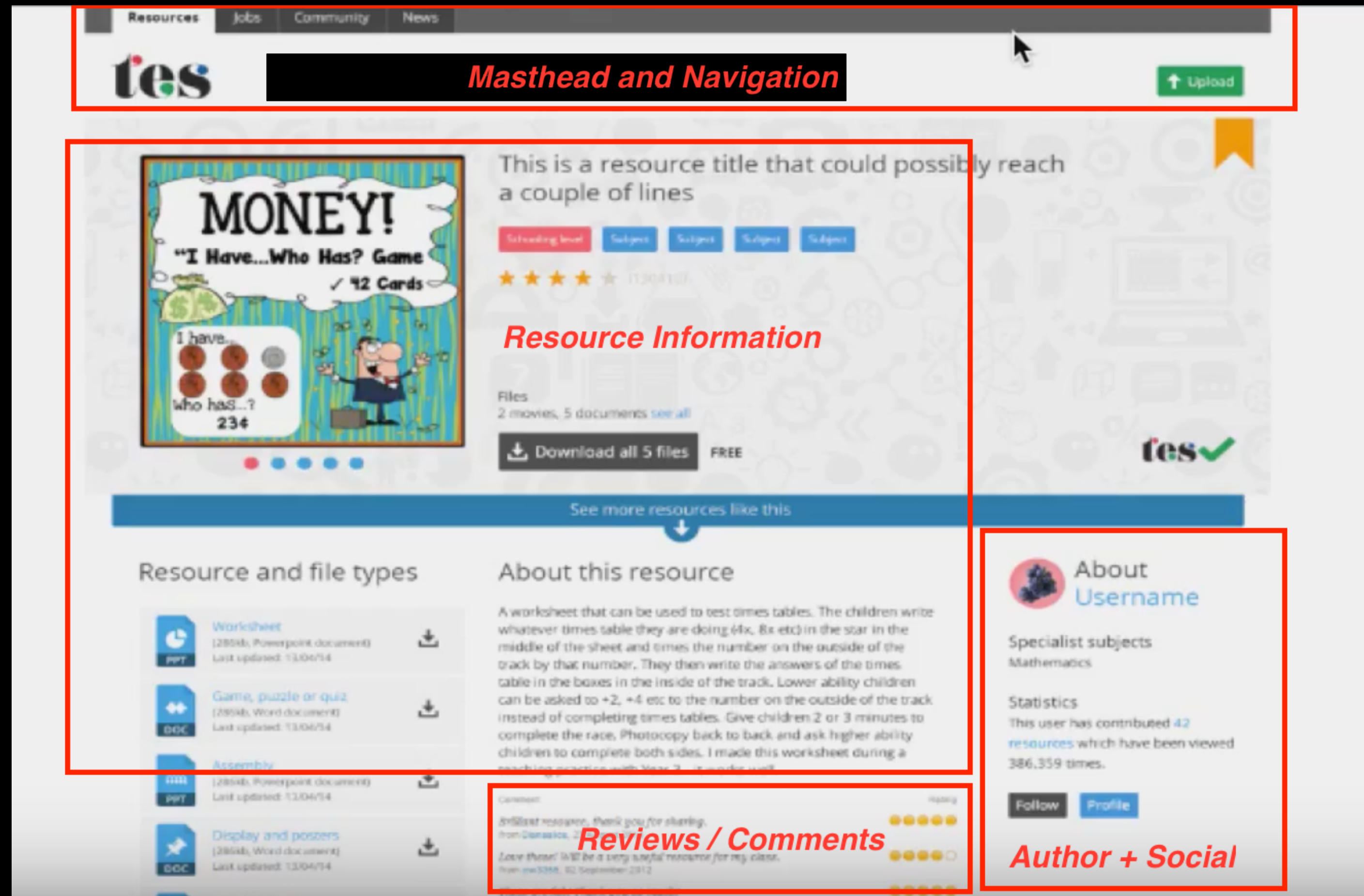
About Username

Specialist subjects
Mathematics

Statistics
This user has contributed [42 resources](#) which have been viewed 386,359 times.

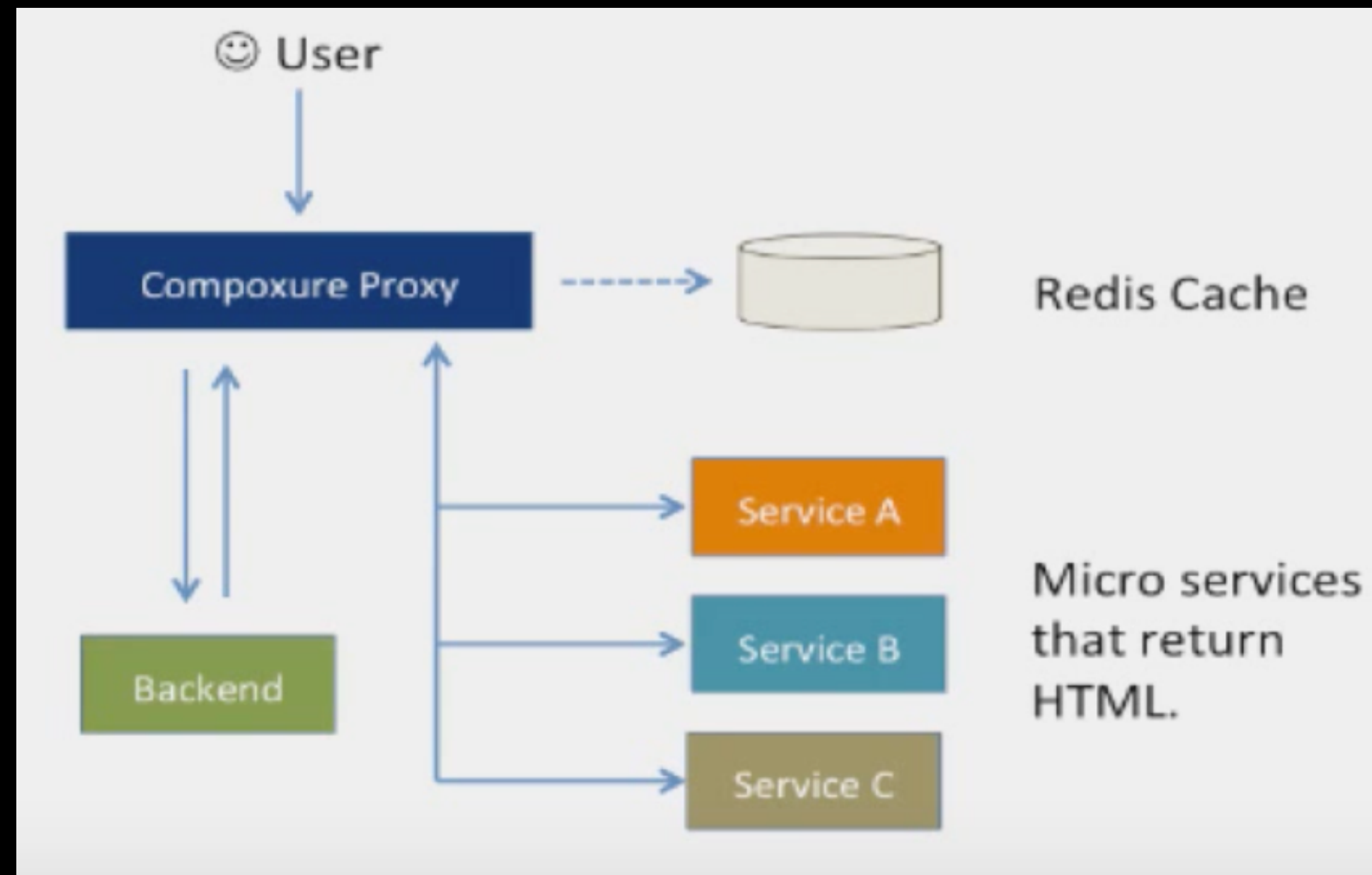
[Follow](#) [Profile](#)

microservices



https://www.youtube.com/watch?v=_SoPZS7Wqiw

one way to implement



implementation by TES Global

links

- <http://redux.js.org>
- <http://alt.js.org>
- <https://github.com/airbnb/enzyme>
- <https://egghead.io/series/react-testing-cookbook>
- <https://egghead.io/series/getting-started-with-redux>
- <https://github.com/uldissturms/talks>